OPTIMASI ALGORITMA ALGA UNTUK MENINGKATKAN LAJU KONVERGENSI

Hari Santoso [1)], Lukman Fakih Lidimilah [2)]
[1] Manajemen Informatika, AMIK Ibrahimy Situbondo
email: haripinter@gmail.com
[2] Manajemen Informatika, AMIK Ibrahimy Situbondo
email: luky.lukman7@gmail.com

Abstact

Artificial Algae Algorithm (AAA) is an optimization algorithm that has advantages of swarm algorithm model and evolution model. AAA consists of three phases of helical movement phase, reproduction, and adaptation. Helical movement is a three-dimensional movement with the direction of x, y, and z which is very influential in the rate of convergence and diversity of solutions. Helical motion optimization aims to increase the convergence rate by moving the algae to the best colony in the population. Algae Algorithm Optimization (AAA ') was tested with 25 objective functions of CEC'05 and implemented in case of pressure vessel design optimization. The results of the CEC'05 function test show that there is an increase in convergence rate at AAA ', but at worst condition of AAA' becomes less stable and trapped in local optima. The complexity analysis shows that AAA has the complexity of O (M3N2O) and AAA 'has the complexity of O (M2N2O) with M is the number of colonies, N is the number of algae individuals, and O is the maximum of the evaluation function. The results of the implementation of pressure vessel design optimization show that AAA's execution time increased 1,103 times faster than AAA. The increase in speed is due to the tournament selection process in AAA performed before the helical motion, whereas in AAA 'is done if the solution after movement is no better than before. At its best, AAA 'found a solution 4.5921 times faster than AAA. At worst, AAA 'stuck on local optima because helical movement is too focused on global best that is not necessarily global optima.

Keywords: *artificial algae algorithm, optimization, convergence rate, pressure vessel design*

1. PENDAHULUAN

Peran kecerdasan komputasional sangat penting sebagai alat bantu dalam pencarian solusi dari berbagai permasalahan kompleks di berbagai bidang, seperti penyelesaian kasus *Optimal Power Flow* (OPF) pada sistem kelistrikan [1], analisis pembebanan ekonomis pada pembangkit termis [2], sistem pendeteksi penyusup jaringan komputer [3], pengontrol kecepatan DC/DC *converter* [4], dan masalah kompleks lainnya. Permasalahan kompleks tersebut seringkali memiliki banyak kandidat solusi sehingga dibutuhkan algoritma yang mampu memilih solusi optimal yang dikenal dengan algoritma optimasi [5].

Hingga saat ini, penelitian terus dilakukan untuk memperoleh algoritma optimasi dengan performa yang baik sebab setiap tidak semua masalah dapat diselesaikan dengan satu jenis algoritma [6]. Perkembangan algoritma optimasi sudah sampai pada model *hybrid* [7], yaitu kombinasi antara model *swarm* dan model evolusi [8][9][10][11][12][13], salah satunya adalah Algoritma Alga (*Artificial Algae Algorithm*, AAA). Algoritma Alga [14] adalah algoritma yang terinspirasi perilaku alga dalam bergerak, berkembang biak, dan beradaptasi dengan lingkungan. AAA lebih stabil daripada *Artificial Bee Colony* (ABC), *Differencial Evolution* (DE), *Ant Colony Optimization* (ACO), dan *Harmony Search* (HS). Meskipun AAA lebih stabil, gerakan alga yang dilakukan secara acak mempengaruhi laju konvergensinya.

Penelitian ini dimaksudkan untuk mengoptimasi gerakan heliks pada AAA sehingga AAA memiliki laju konvergensi lebih baik. Gerakan heliks AAA terinspirasi oleh perilaku alga pada kehidupan nyata. Koloni alga mencerminkan kandidat solusi, individu alga mencerminkan dimensi ruang solusi, dan sumber cahaya sebagai target solusi. Pada AAA versi standar, alga bergerak secara acak dan hanya dipengaruhi oleh koloni tetangga (*local best*). Menurut peneliti, koloni alga akan cenderung bergerak ke arah target solusi yang paling baik, yaitu koloni yang mendapatkan cahaya paling kuat dalam populasi. Dalam hal ini AAA dengan gerakan heliks menuju cahaya terkuat disebut dengan AAA' karena gerakan alga pada AAA' tidak hanya terpengaruh oleh koloni terbaik tetangga (*local best*) tetapi juga bergerak menuju koloni terbaik dalam populas (*global best*).

Hasil optimasi akan ditentukan dengan membandingan AAA dan AAA' berdasarkan pengujian 25 fungsi dari CEC'05 dan implementasi pada kasus optimasi desain bejana tekan *(pressure vessel)* [15]. Performa ditentukan berdasarkan laju konvergensi dan kompleksitas waktu dalam menemukan solusi. Dengan optimasi ini, diharapkan terjadi peningkatan pada performa Algoritma Alga sehingga dapat menjadi alternatif dalam pencarian solusi dari permasalahan kompleks dalam kehidupan sehari-hari.

2. METODE PENELITIAN

2.1 Algoritma Alga

Algoritma Alga merupakan algoritma optimasi yang terinspirasi oleh cara alga bergerak, berkembang biak, dan beradaptasi. Algoritma Alga mengombinasikan model evolusi dan model *swarm*. Algoritma ini merupakan algoritma dengan pendekatan probabilistik dan mampu menyelesaikan masalah *global optimization*. Algoritma Alga terdiri dari tiga tahap yaitu pergerakan heliks, evolusi, dan adaptasi.

A. Tahap Gerakan Heliks

Alga adalah sel tunggal yang memiliki klorofil sebagai penghasil energi sehingga alga akan bergerak mendekati cahaya untuk menghasilkan energi. Alga bergerak dengan cara melingkar seperti spiral yang disebut dengan gerakan heliks. Gerakan heliks adalah gerakan 3 dimensi, yaitu pada sumbu x, y dan z. Setiap alga bergerak, alga akan kehilangan energi karena bergesekan dengan cairan sekitar sehingga energi yang dihasilkan juga akan berkurang selama perjalanan. Gerakan heliks ditentukan dengan Persamaan (1-3).

$$x_{im}^{t+1} = x_{im}^{t} + \left(x_{jm}^{t} - x_{im}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)p \quad (1)$$

$$x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)\cos\alpha \quad (2)$$

$$x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)\sin\beta \quad (3)$$

Pada persamaan (1-3), x^t_i adalah koloni alga ke-*i* pada waktu ke-*t*. Koloni tetangga x_i dipilih dengan *tournament selection* dengan syarat $x_j \neq x_i$ dan individu alga $k \neq l \neq m$. Parameter Δ adalah gaya gesek cairan, sedangkan $\tau^t(x_i)$ adalah gaya gesek yang dialami alga ke-*i*. Persamaan (1) adalah gerakan pada sumbu z, nilai *p* dipilih acak [-1,1]. Persamaan (2) adalah gerakan pada sumbu x dengan α dipilih acak [0,2π]. Persamaan (3) adalah gerakan pada sumbu y dengan β dipilih acak [0, 2π].

B. Tahap Evolusi

Koloni alga yang kehabisan energi akan mati, sedangkan koloni alga yang kuat akan berkembang biak dengan membelah diri. Kebugaran koloni alga ditentukan berdasarkan laju perkembang biakannya. Laju perkembang biakan alga μ ditentukan dengan Persamaan (4) yang ditentukan dengan laju perkembangbiakan maksimum μ_{max} pada satuan waktu (1/waktu), kondisi alga S, dan K_s konstanta cairan (massa/volume).

$$\mu = \frac{\mu_{max} S}{K_S + S} \quad (4)$$

Koloni yang memiliki kebugaran tinggi akan bertahan hidup sedangkan koloni dengan kebugaran rendah akan mati. Tingkat kebugaran koloni *G* dihitung dengan Persamaan (5) dan proses evolusi ditentukan pada Persamaan (6-8) dengan *biggest* adalah koloni terbugar, *smallest* koloni terlemah, dan *m* adalah individu alga.

$$G_i^{t+1} = \mu_i^t G_i^t,\ i=1, 2, ..., N \quad (5)$$

$$biggest^t = max\ G_i^t,\ i = 1, 2, ..., N \quad (6)$$

$$smallest^t = min\ G_i^t,\ i = 1, 2, ..., N \quad (7)$$

$$smallest_m^t = biggest_m^t,\ m = 1, 2, ..., G \quad (8)$$

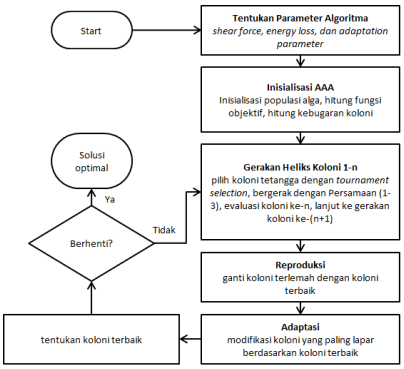
C. Tahap Adaptasi

Ketika koloni alga melemah, maka alga akan mencoba beradaptasi dengan lingkungan dengan menyesuaikan diri pada alga terbugar. Alga yang lemah diindikasikan dengan tingkat kelaparan yang tinggi. Semakin jauh alga dari sumber cahaya dan semakin lama alga bergerak akan menambah tingkat kelaparan alga. Tingkat kelaparan alga *starving* ditentukan berdasarkan kebugaran alga yang didefinisikan dengan persamaan (9-10).

$$starving^t = max\ A_i^t,\ i = 1, 2, ..., N \quad (9)$$

$$starving^{t+1} = starving^t + (biggest^t + starving^t)\ x\ rand \quad (10)$$

D. Flowchart dan Pseudocode Algoritma Alga



Gambar 1. *Flowchart* AAA versi standar

Fungsi objektif untuk optimasi f(x), x=(x1,x2,...,xd)
Inisialisasi populasi n koloni alga secara random
Hitung tingkat kebugaran (G) dari n koloni
Tentukan parameter algoritma (shear force Δ, loss of energy e dan Adaptation parameter Ap)
***While** (t < MaxCalculation)*
  Hitung Energi (E) dan gaya gesek koloni (τ) dari n koloni
  ***For** i=1:n*
    Tingkat kelaparan koloni (starving) = true
    ***While**($E(x_i) > 0$)*
      Pilih koloni alga j dengan tournament selection
      Pilih tiga individu alga untuk pergerakan heliks (k, l, m)
      $x_{im}^{t+1} = x_{im}^{t} + (x_{jm}^{t} - x_{im}^{t})(\Delta - \tau^{t}(x_i))p$
      $x_{ik}^{t+1} = x_{ik}^{t} + (x_{jk}^{t} - x_{ik}^{t})(\Delta - \tau^{t}(x_i))\cos\alpha$
      $x_{il}^{t+1} = x_{il}^{t} + (x_{jl}^{t} - x_{il}^{t})(\Delta - \tau^{t}(x_i))\sin\beta$
      Sudut α,β random[0,2π] dan nilai p juga random [-1,1]
      Hitung solusi sesuai fungsi optimasi
      $E(x_i)$ = $E(x_i)$ - (e/2) energi dipakai untuk bergerak
      ***if** solusi lebih baik dari sebelumnya*
        Ganti koloni i dan ganti tingkat kelaparan i = false
      else
        $E(x_i)$=$E(x_i)$-(e/2) energi dipakai untuk metabolisme
      end if
    end While
    ***if** Tingkat kelaparan = true*
      Naikkan nilai starving $A(x_i)$
    end if
  end For
  Hitung tingkat kebugaran (G) seluruh populasi
  Pilih salah satu dimensi (individu alga) r untuk reproduksi
  $smallest^{t}_{r}$ = $biggest^{t}_{r}$
  ***if** rand < Ap*
    starving $^{t+1}$ = starving^{t} + (biggest^{t} - starving^{t}) x rand
  end if
  pilih koloni terbaik sebagai solusi
end While

Gambar 2. *Pseudocode* AAA versi standar

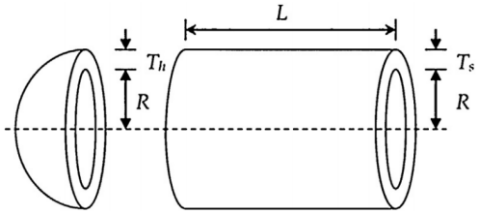
2.2 Pengujian 25 Fungsi Objektif CEC'05

Pada dataset CEC'05 terdapat 25 jenis fungsi objektif yang terdiri dari 5 fungsi *unimodal* dan 20 fungsi *multimodal* dengan tiga jenis fungsi yaitu *basic function*, *expanded function*, dan *hybrid composition function*. Performa algoritma ditentukan oleh tingkat kesalahan (*error*), akurasi, tingkat keberhasilan (*success rate* dan *success performance*), laju konvergensi, dan kompleksitas algoritma.

Tingkat kesalahan (*error*) terkait dengan selisih antara hasil yang diperoleh algoritma dengan target solusi. Akurasi dan tingkat keberhasilan (*success rate* dan *success performance*) terkait dengan kemampuan algoritma menemukan solusi atau tidak. Laju konvergensi menggambarkan grafik perbandingan antara semilog selisih solusi dan target solusi dengan fungsi evaluasi. Kompleksitas algoritma terkait kerumitan algoritma ditinjau dari *execution time*.

2.3 Permasalahan Optimasi Desain Pressure Vessel

Pada penelitian ini, tujuan dari permasalahan desain *pressure vessel* adalah untuk menentukan ukuran dan dimensi *pressure vessel* dengan volume 21.24 m^3 dan bekerja pada tekanan 3000 psi (20.68 MPa) dengan *cost* yang rendah. Desain *pressure vessel* tampak pada Gambar 3 berbentuk silinder dengan masing-masing tutup ujungnya berbentuk setengah lingkaran. Hasil optimasi ditentukan oleh empat parameter, Ts (x_1) dan Th (x_2) masing-masing bernilai 1.125 inci dan 0.0625 inci, sedangkan jari-jari dalam bejana R (x_3) dan panjang silinder L (x_4) tidak ditentukan nilainya.



Gambar 3. Skema desain *pressure vessel* dengan ketebalan Ts dan Th, diameter dalam R dan panjang silinder L

Peneliti sebelumnya telah melakukan penelitian tentang desain *pressure vessel* dengan berbagai parameter dan *constrain*. Hasil dari penelitian tersebut menyimpulkan bahwa *range* parameter, *constrain*, dan fungsi minimasi ditentukan dengan ketentuan berikut:

Parameter :
$x_1 \in [1.125, 12.5]$ *granularity* 0.0625
$x_2 \in [0.625, 12.5]$ *granularity* 0.0625
$x_3 \in [0.240]$
$x_4 \in [0.240]$

Constrain :
$g1(x) = 0.0193x_3 - x_1 \le 0$
$g2(x) = 0.0095x_3 - x_2 \le 0$
$g_3(x) = 750.0 \times 1728.0 - \pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 \le 0$
$g4(x) = x_4 - 240.0 \le 0$
$g5(x) = 1.1 - x_1 \le 0$
$g6(x) = 0.6 - x_2 \le 0$

Fungsi minimasi :
$f(x) = 0.06224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1611x_1^2x_4 + 19.84x_1^2x_3$

Nilai optimum yang diperoleh yaitu $f(x)$ = 7197.72893 dengan nilai parameter x_1 = 1.125,

x_2 = 0.625, x_3 = 58.2901554, serta x_4 = 43.6926562.

2.4 Optimasi Gerakan Heliks

Optimasi gerakan heliks pada AAA' didasarkan pada perilaku alami alga, yaitu koloni alga yang lebih dekat dengan cahaya akan memiliki kebugaran lebih baik dari alga yang jauh dari cahaya. Dengan demikian, setiap koloni alga akan cenderung mendekat dengan koloni alga terbaik untuk mendapatkan cahaya lebih banyak. Dalam hal ini, koloni terbaik dalam populasi disebut dengan *global best*, sedangkan koloni terbaik tetangga disebut dengan *local best*. Untuk mengoptimasi gerakan heliks sesuai perilaku tersebut, maka tiga tahap yang harus dilakukan yaitu pemilihan *local best* dan *global best*, gerakan heliks, serta pengondisian gerakan alga.

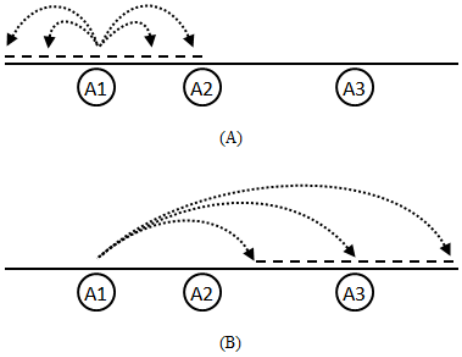
A. Pemilihan Global Best dan Local Best

Pemilihan koloni alga sebagai *local best* atau *global best* ditentukan setelah perhitungan fungsi objektif (minimasi atau maksimasi) yaitu saat inisialisasi koloni, setelah fase gerakan heliks, dan setelah fase adaptasi. Pemilihan *local best* dan *global best* saat inisialisasi dan setelah fase adaptasi dilakukan dengan memilih koloni terbaik sebagai *local best* dan *global best*. Sedangkan pemilihan *local best* dan *global best* setelah gerakan heliks ditentukan dengan 3 kriteria:

- Jika alga menjadi lebih baik setelah gerakaan heliks, maka jadikan sebagai koloni terbaik tetangga (*local best*);
- Jika alga menjadi yang terbaik setelah gerakan heliks, maka jadikan alga sebagai koloni terbaik (*global best*);
- Jika alga tidak lebih baik setelah gerakan heliks, maka *local best* dipilih dengan *tournament selection*.

B. Gerakan Heliks

Berdasarkan gerakan yang mereferensi pada koloni alga terbaik (*global best*) dan koloni terbaik tetangga (*local best*), maka gerakan heliks hasil optimasi akan cenderung bergerak menuju posisi *global best*. Ilustrasi gerakan tampak pada Gambar 4, A1 adalah alga yang sedang bergerak, A2 *local best* dan A3 *global best*. Secara komputasi, gerakan heliks pada AAA tampak seperti Gambar 4A menunjukkan area gerakan berada di sekitar posisi awal alga sejauh koloni tetangga (*local best*) yang terpilih. Setelah optimasi, gerakan alga tampak seperti Gambar 4B yang menunjukkan gerakan alga akan mengarah ke sekitar posisi alga terbaik (*global best*).



Gambar 4. Gerakan heliks AAA versi standar (A) dan setelah optimasi (B)

Berdasarkan rancangan gerakan pada Gambar 4B, gerakan heliks setelah optimasi didefinisikan pada Persamaan (11-14).

$$x_c^t = \frac{x_B + x_b}{2} \quad (11)$$

$$x_{im}^{t+1} = x_{Bm}^t + (x_{cm}^t - x_{im}^t)(\Delta - \tau^t(x_i))p \quad (12)$$

$$x_{ik}^{t+1} = x_{Bk}^t + (x_{ck}^t - x_{ik}^t)(\Delta - \tau^t(x_i))\cos\alpha \quad (13)$$

$$x_{il}^{t+1} = x_{Bl}^t + (x_{cl}^t - x_{il}^t)(\Delta - \tau^t(x_i))\sin\beta \quad (14)$$

Perubahan dilakukan pada x_B, x_b, dan x_c dengan x_B adalah *global best*, x_b adalah *local best*, dan x_c rata-rata antara *global best* dan *local best*. Koloni terbaik x_B dan koloni terbaik tetangga x_c berubah selama perulangan *t* dengan dimensi *k*, *l*, dan *m* yang merepresentasikan individu dalam koloni tersebut.

C. Pengondisian Gerakan

Sebagaimana disebutkan bahwa gerakan heliks sangat berpengaruh pada keragaman solusi dan laju konvergensi, maka diperlukan aturan untuk mengondisikan gerakan agar AAA tidak terjebak pada *local optima*. Berdasarkan Gambar 4, gerakan alga akan meningkatkan daya eksploitasi terhadap solusi terbaik dan mengabaikan kandidat solusi lainnya. Oleh sebab itu perlu pembatasan agar gerakan alga tidak hanya menguat untuk mengeksploitasi, tetapi juga untuk eksplorasi terhadap kandidat solusi lainnya. Untuk membatasi daya eksploitasi, perilaku berikut dapat diterapkan pada AAA:

- Jika alga lebih baik setelah gerakan heliks, energi yang dimiliki bertambah karena lebih dekat dengan cahaya.
- Jika alga menjadi yang terbaik, maka alga harus diam. Perilaku ini bertujuan agar

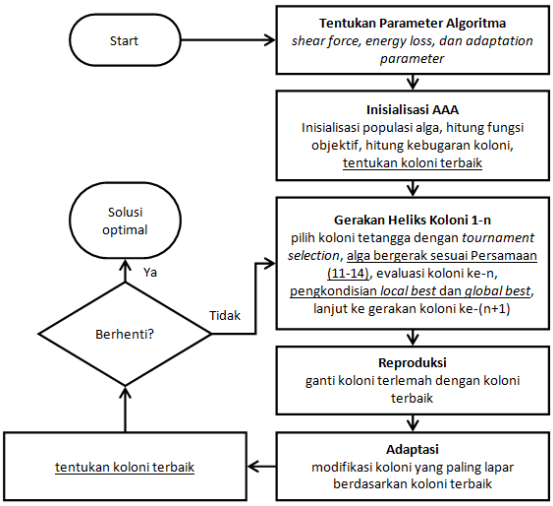
alga lain tidak terjebak jika solusi yang ditemukan adalah *local optima*.

- Jika alga tidak lebih baik setelah gerakan heliks, maka alga terus bergerak hingga energinya habis. Tujuannya untuk meningkatkan daya eksplorasi kandidat solusi yang jauh dari solusi.

Perilaku pengondisian tersebut diterapkan saat pemilihan *global best* dan *local best* seperti tampak dalam *pseudocode* pada Gambar 5. *Flowchart* AAA' dapat dilihat pada Gambar 6. Keterangan yang digaris bawah adalah bagian yang bertambah atau berubah setelah optimasi.

Pilih $Alga^t_i$
$Alga_i^t$ *melakukan gerakan heliks*
JIKA solusi($Alga_i^t$) lebih baik dari solusi($Alga_i^{t-1}$)
    Local_Best = $Alga_i^t$
    Tingkatkan nilai $Energi\_Alga_i^t$ *agar lebih lama bergerak*
    JIKA solusi($Alga_i^t$) lebih baik dari solusi(Global_Best)
        Global_Best = $Alga_i^t$
        Hilangkan $Energi\_Alga_i^t$ *agar berhenti bergerak*
JIKA TIDAK
    Local_Best dipilih dengan *tournament selection*

Gambar 5. *Pseudocode* kriteria pemilihan *global best* dan *local best* serta pengondisian gerakan alga



Gambar 6. *Flowchart* Algoritma Alga setelah optimasi (AAA')

3. HASIL DAN PEMBAHASAN

Pengujian dalam penelitian ini dilakukan dengan membandingkan hasil solusi AAA dan AAA' berdasarkan kasus minimasi desain *pressure vessel*. Spesifikasi komputer yang digunakan yaitu *processor* Core2 Duo 1.6 GHz, RAM 1 x DDR2 2 GB, dan *harddisk* Seagate 250 GB. *Software* yang digunakan adalah Matlab 2013a dengan bilangan random *uniform* menggunakan algoritma *Substract With Borrow* (SWB). Konfigurasi Algoritma Alga yaitu 40 koloni, setiap koloni terdiri dari 4 individu alga yang mewakili $x_{1\text{-}4}$, konstanta gaya gesek cairan Δ=2, parameter pengurangan energi e = 0.3, dan parameter adaptasi Ap = 0.3.

Pada pengujian CEC'05, pengujian dilakukan dengan varasi dimen 10, 30, dan 50 dengan maksimum fungsi evaluasi 100000, 300000, dan 500000. Masing-masing fungsi dijalankan 25 kali dengan *random state* 1-25. Berdasarkan hasil pengujian, nilai *error* untuk dimensi 10, 30, dan 50 bisa dilihat pada Tabel 1, 2, dan 3.

Tabel 1. Perbandingan kesalahan (error) antara AAA dan AAA' pada fungsi F1-F25 dengan D=10 dan fungsi evaluasi 100000

F	AAA	AAA'	F	AAA	AAA'
F1	0	0	F14	2.532635e+00	**1.845908e+00**
F2	3.544051e-08	**0**	F15	0	0
F3	4.332011e+04	**1.584129e+04**	F16	1.062735e+02	**1.021275e+02**
F4	1.213441e-04	**0**	F17	1.111575e+02	**1.014830e+02**
F5	0	0	F18	300	300
F6	**6.524703e-03**	1.865608e-02	F19	300	300
F7	1.267046e+03	1.267046e+03	F20	300	300
F8	**2.001468e+01**	2.010727e+01	F21	500	**300**
F9	0	0	F22	**300**	7.321192e+02
F10	6.302890e+00	**5.969754e+00**	F23	5.594683e+02	5.594683e+02
F11	1.837553e+00	**1.153795e+00**	F24	200	200
F12	**5.375462e-01**	1.071656e+01	F25	8.122389e+02	**5.983106e+02**
F13	2.673952e-01	**1.700844e-01**			

Tabel 2. Perbandingan kesalahan (error) antara AAA dan AAA' pada fungsi F1-F25 dengan D=30 dan fungsi evaluasi 300000

F	AAA	AAA'	F	AAA	AAA'
F1	5.684342e-14	5.684342e-14	F14	1.164699e+01	**1.090864e+01**
F2	2.683738e+00	**2.172347e-02**	F15	0	0
F3	**1.121913e+06**	1.814886e+06	F16	9.324127e+01	**7.715226e+01**
F4	2.823876e+03	**1.654992e+03**	F17	1.177208e+02	**7.974424e+01**
F5	**2.272797e+03**	2.302074e+03	F18	**9.050730e+02**	9.081688e+02
F6	**2.780308e-03**	5.754464e+00	F19	**9.048526e+02**	9.056769e+02
F7	4.696289e+03	4.696289e+03	F20	**8.000000e+02**	9.077718e+02
F8	**2.004667e+01**	2.055709e+01	F21	5.000000e+02	5.000000e+02
F9	5.684342e-14	5.684342e-14	F22	**8.696635e+02**	8.706625e+02
F10	7.263188e+01	**4.818957e+01**	F23	5.341641e+02	5.341641e+02
F11	**1.502232e+01**	2.406120e+01	F24	2.000000e+02	2.000000e+02
F12	**1.400096e+03**	2.484439e+04	F25	9.794573e+02	**9.682045e+02**
F13	1.083688e+00	**8.321185e-01**			

Tabel 3. Perbandingan kesalahan (error) antara AAA dan AAA' pada fungsi F1-F25 dengan D=50 dan fungsi evaluasi 500000

F	AAA	AAA'	F	AAA	AAA'
F1	1.136868e-13	**5.684342e-14**	F14	2.174806e+01	**2.073107e+01**
F2	8.573266e+01	**4.107799e+01**	F15	7.048558e+00	**3.684604e+00**
F3	**2.215416e+06**	3.667225e+06	F16	1.932701e+02	**1.070166e+02**
F4	**2.354905e+04**	2.681825e+04	F17	1.664748e+02	**1.234143e+02**
F5	7.351880e+03	**6.970583e+03**	F18	**9.212163e+02**	9.263443e+02
F6	**3.878069e-02**	3.289907e+01	F19	**8.000000e+02**	9.243677e+02
F7	6.195318e+03	6.195318e+03	F20	**9.211946e+02**	9.254899e+02
F8	**2.006642e+01**	2.037659e+01	F21	5.000000e+02	5.000000e+02
F9	**1.136868e-13**	1.705303e-13	F22	9.451303e+02	**8.986300e+02**
F10	2.410505e+02	**9.192878e+01**	F23	5.391222e+02	5.391222e+02
F11	**4.120773e+01**	4.885721e+01	F24	2.000000e+02	2.000000e+02
F12	**8.595387e+03**	1.537204e+05	F25	1.262342e+03	**1.232424e+03**
F13	2.799388e+00	**1.952946e+00**			

Berdasarkan Tabel 1, 2, dan 3, pada dimensi yang berbeda AAA cenderung lebih stabil daripada AAA' sebab dari dimensi rendah ke tinggi, fungsi yang mampu diselesaikan oleh AAA meningkat. AAA' memiliki kecepatan dalam menemukan solusi yaitu pada dimensi 10, AAA mampu menyelesaikan 11 fungsi lebih baik dari AAA. Akan tetapi, fungsi yang mampu diselesaikan lebih baik dari AAA turun naik yang mengindikasikan bahwa AAA' kurang stabil.

Tabel 4. Perbandingan Akurasi dan Tingkat Keberhasilan (*Success Rate*) Antara AAA dan AAA' pada Dimensi 10

Fungsi	Alg.	Best	Mean	Std	Success Rate	Success Performance
1	AAA	8030	8.676240e+03	3.105445e+02	100%	8.676240e+03
	AAA'	**1785**	2.131560e+03	1.823605e+02	100%	2.131560e+03
2	AAA	63971	7.165084e+04	4.715393e+03	100%	7.165084e+04
	AAA'	**17126**	2.123888e+04	2.281446e+03	100%	2.123888e+04
3	AAA	-	-	-	0%	-
	AAA'	-	-	-	0%	-
4	AAA	-	-	-	0%	-
	AAA'	**27873**	3.434124e+04	4.365355e+03	100%	3.434124e+04
5	AAA	28224	37392	6.546338e+03	100%	37392
	AAA'	**3852**	9.064200e+03	2.859775e+03	100%	9.064200e+03
6-8	AAA	-	-	-	0%	-
	AAA'	-	-	-	0%	-
9	AAA	16017	1.809628e+04	1.222460e+03	100%	1.809628e+04
	AAA'	**6160**	1.321276e+04	8.338223e+03	84%	1.572948e+04
10-14	AAA	-	-	-	0%	-
	AAA'	-	-	-	0%	-
15	AAA	23443	2.715180e+04	3.164469e+03	40%	6.787950e+04
	AAA'	**9106**	1.763143e+04	9.372224e+03	28%	6.296939e+04
16-25	AAA	-	-	-	0%	-
	AAA'	-	-	-	0%	-

Tabel 5. Perbandingan Akurasi dan Tingkat Keberhasilan (*Success Rate*) Antara AAA dan AAA' pada Dimensi 30

Fungsi	Alg.	Best	Mean	Std	Success Rate	Success Performance
1	AAA	26897	2.898704e+04	9.588133e+02	100%	2.898704e+04
	AAA'	**7649**	8.331600e+03	3.953127e+02	100%	8.331600e+03
2-8	AAA	-	-	-	-	-
	AAA'	-	-	-	-	-
9	AAA	49856	5.771604e+04	1.018365e+04	96%	6.012088e+04
	AAA'	**29337**	5.334320e+04	2.646360e+04	60%	8.890533e+04
10-14	AAA	-	-	-	-	-
	AAA'	-	-	-	-	-
15	AAA	74797	7.623650e+04	2.035760e+03	8%	9.529563e+05
	AAA'	**66012**	66012	0	4%	1650300
16-25	AAA	-	-	-	-	-
	AAA'	-	-	-	-	-

Tabel 6. Perbandingan Akurasi Dan Tingkat Keberhasilan (*Success Rate*) Antara AAA Dan AAA' pada Dimensi 50

Fungsi	**Alg.**	**Best**	**Mean**	**Std**	**Success Rate**	**Success Performance**
1	AAA	47561	5.029632e+04	1.180371e+03	100%	5.029632e+04
	AAA'	**14148**	1.582780e+04	6.457945e+02	100%	1.582780e+04
2-8	AAA	-	-	-	-	-
	AAA'	-	-	-	-	-
9	AAA	**82477**	9.012858e+04	4.209324e+03	48%	1.877679e+05
	AAA'	186956	186956	0	4%	4673900
10-25	AAA	-	-	-	-	-
	AAA'	-	-	-	-	-

Akurasi dan tingkat keberhasilan menunjukkan bahwa apakah algoritma mampu menemukan solusi sebelum maksimum fungsi evaluasi atau tidak. Perbandingan akurasi AAA dan AAA' dengan dimensi 10, 30, dan 50 dapat dilihat pada Tabel 4, 5, dan 6. Berdasarkan Tabel 4, 5, dan 6 yaitu pada dimensi 10, dari 25 kali pengujian F1, AAA dan AAA' sama-sama memiliki *success rate* 100%, artinya 25 kali pengujian berhasil dan menghasilkan solusi dengan *error* yang kecil. Akan tetapi, AAA' lebih dulu menemukan solusi yaitu pada fungsi evaluasi ke-1785 sedangkan AAA menemukan solusi pada fungsi evaluasi ke-8030 yang berarti AAA' lebih cepat menemukan solusi.

Success performance menunjukkan performa rata-rata terkait fungsi evaluasi yang dibutuhkan untuk menemukan solusi pada fungsi tersebut. Pada dimensi 10 untuk F1, AAA memiliki *success performance* sekitar 8676 yang berarti nilai tersebut adalah nilai fungsi evaluasi standar untuk menemukan solusi khusus untuk F1. Sedangkan untuk AAA' memiliki *success performance* lebih rendah yaitu sekitar 2132 yang berarti lebih baik dari AAA karena membutuhkan nilai fev yang lebih kecil. Pada fungsi F3, F4, F6-8, F10-14, F16-25 masing-masing AAA dan AAA' tidak menemukan solusi yang baik. Sehingga nilai *success rate* dan *success rate* kosong.

Berdasarkan Tabel 4, 5, dan 6, secara umum AAA memiliki *success rate* yang lebih tinggi dari AAA' pada setiap dimensi. Akan tetapi, AAA' memiliki *success performance* yang lebih baik karena mampu menemukan solusi lebih cepat, meski hasil didapat kurang stabil sebagaimana ditunjukkan pada fungsi F15 pada dimensi 10 dan 30. Pada dimensi 10, AAA memiliki *success rate* 40% sedangkan AAA' 28%. Namun, *success performance* AAA' lebih baik yaitu 6297 sedangkan AAA adalah 6788. Pada kondisi terbaik, AAA' menemukan solusi pada fungsi evaluasi ke-9106 sedangkan AAA menemukan solusi pada fungsi evaluasi ke-23443. Namun pada dimensi 30, *success performance* AAA lebih baik dari AAA' meski AAA' menemukan solusi lebih cepat. Hal ini memperkuat asumsi bahwa AAA lebih stabil dari AAA'.

Laju konvergensi digambarkan berdasarkan semilog dari perbedaan antara solusi yang dihasilkan algoritma dan target solusi yang diharapkan. Dari 25 pengujian tiap fungsi, yang diambil adalah median dari ke-25 data tersebut. Grafik laju konvergensi AAA dan AAA' untuk F1-25 dengan dimensi 10, 30, dan 50 dapat dilihat pada Gambar 7 – 81.

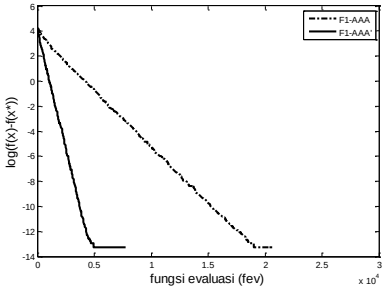
Pada dimensi 10, laju konvergensi AAA' lebih baik dari AAA pada 15 fungsi, yaitu fungsi F1-5, F7, F9-11, F13, F14, F16, F17, F24, dan F25. Sedangkan pada 10 fungsi sisanya, yaitu F6, F5, F12, F15, dan F18-23 laju konvergensi AAA' mengalami penurunan. Bahkan pada fungsi F12, F15, F18-21, dan F23 algoritma AAA' terjebak pada *local optima* yang ditandai dengan garis lurus memanjang yang berarti hasil yang ditemukan oleh AAA' tidak pernah berubah.

Pada dimensi 30, AAA dan AAA' sama-sama terjebak di *local optima* pada fungsi F7, F21, dan F23. Selain itu, AAA' juga terjebak pada *local optima* pada fungsi F6, F15, dan F23. AAA' lebih baik di 13 fungsi yaitu fungsi F2, F4, F7, F10, F13, F14, F16, F17, dan F21-25. Jika dibandingkan dengan dimensi 10, jumlah fungsi yang memiliki laju konvergensi lebih dari AAA menurun dari 15 fungsi menjadi 13 fungsi.

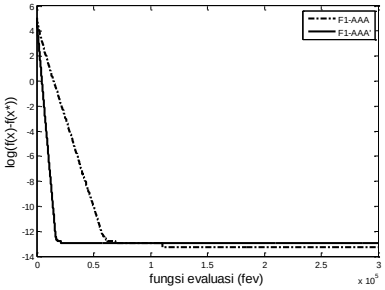
Pada dimensi 50, AAA' lebih baik di 13 fungsi yaitu fungsi F1-2, F4-5, F7, F10, F13-14,

F16, F17, F22, F24, dan F25. Namun AAA' terjebak di *local optima* pada fungsi F7, F9, F15, F18-21, dan F23. Pada dimensi 50, AAA terjebak pada *local optima* pada fungsi F7, F9, F10, F16, F18-21, dan F23. Dengan demikian, AAA dan AAA' menjadi kurang stabil pada dimensi 50 dibandingkan dengan dimensi 10 dan 30.

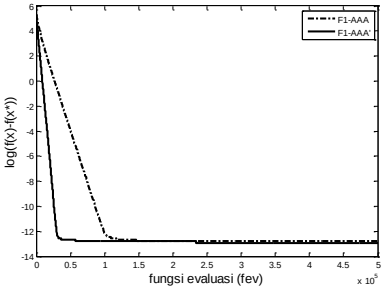
Berdasarkan pengujian pada dimensi 10, 30, dan 50, total fungsi yang mampu diselesaikan AAA' lebih baik dari AAA sejumlah 41 fungsi. Sedangkan laju konvergensi AAA yang lebih baik dari AAA' adalah 34 fungsi. Berdasarkan data tersebut, secara umum AAA' memiliki laju konvergensi yang lebih baik dari AAA.



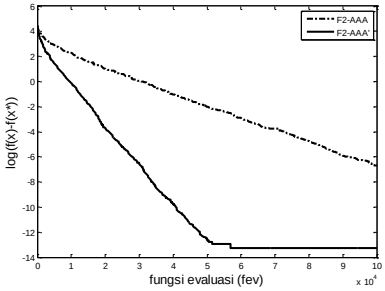
Gambar 7. Grafik laju konvergensi F1 dengan D=10



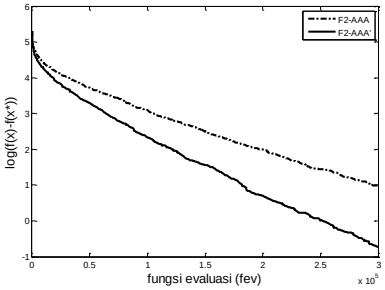
Gambar 8. Grafik laju konvergensi F1 dengan D=30



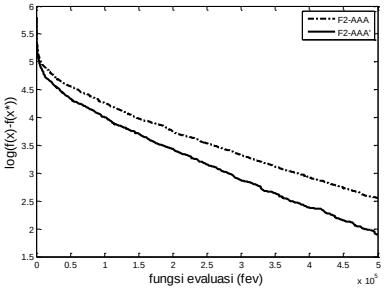
Gambar 9. Grafik laju konvergensi F1 dengan D=50



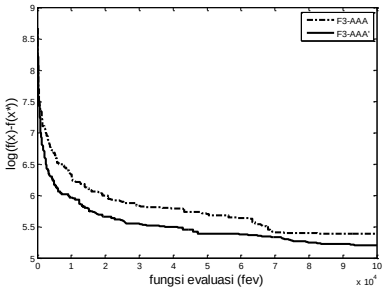
Gambar 10. Grafik laju konvergensi F2 dengan D=10



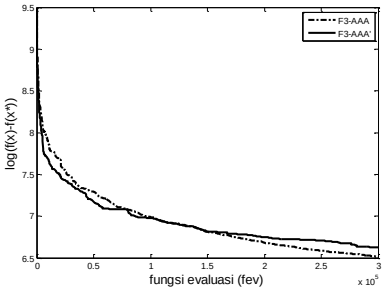
Gambar 11. Grafik laju konvergensi F2 dengan D=30



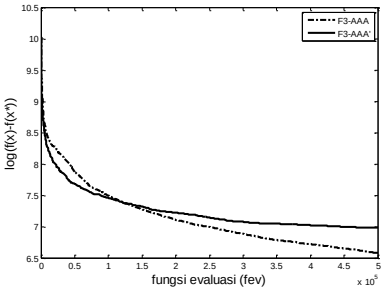
Gambar 12. Grafik laju konvergensi F2 dengan D=50



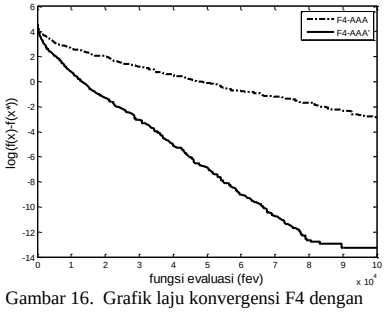
Gambar 13.Grafik laju konvergensi F3 dengan D=10



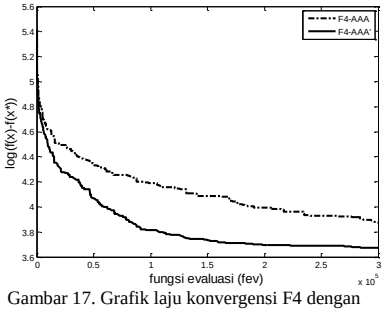
Gambar 14. Grafik laju konvergensi F3 dengan D=30



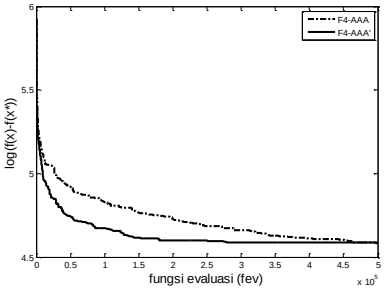
Gambar 15. Grafik laju konvergensi F3 dengan D=50



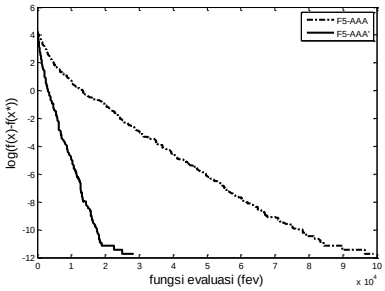
Gambar 16. Grafik laju konvergensi F4 dengan D=10



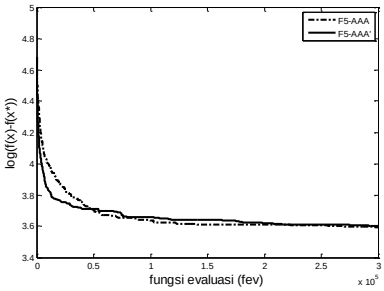
Gambar 17. Grafik laju konvergensi F4 dengan D=30



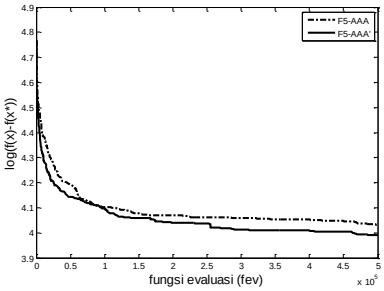
Gambar 18. Grafik laju konvergensi F4 dengan D=50



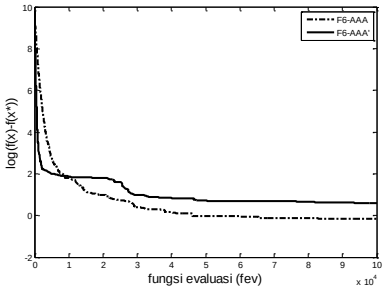
Gambar 19. Grafik laju konvergensi F5 dengan D=10



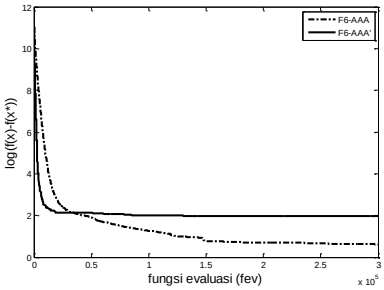
Gambar 20. Grafik laju konvergensi F5 dengan D=30



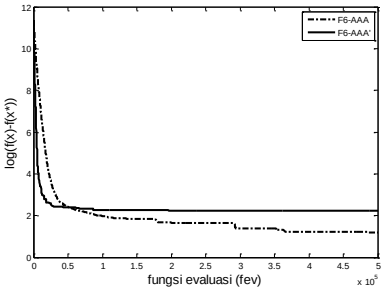
Gambar 21. Grafik laju konvergensi F5 dengan D=50



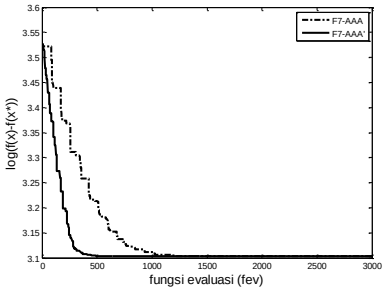
Gambar 22. Grafik laju konvergensi F6 dengan D=10



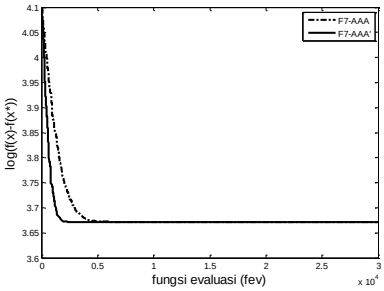
Gambar 23. Grafik laju konvergensi F6 dengan D=30



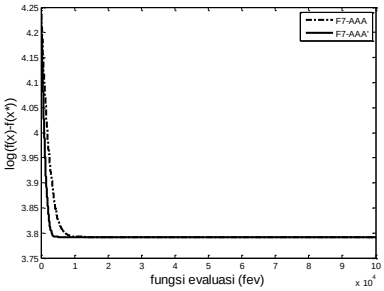
Gambar 24. Grafik laju konvergensi F6 dengan D=50



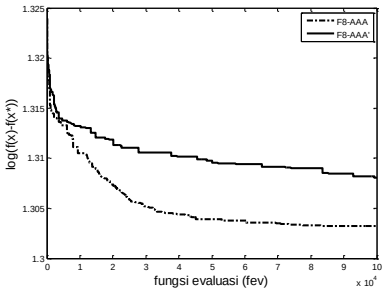
Gambar 25. Grafik laju konvergensi F7 dengan D=10



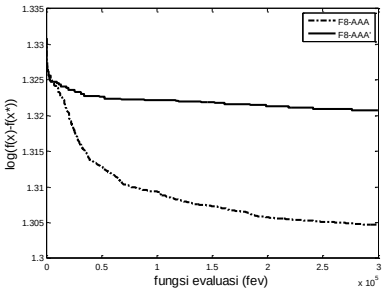
Gambar 26. Grafik laju konvergensi F7 dengan D=30



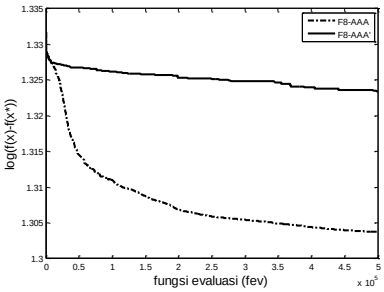
Gambar 27. Grafik laju konvergensi F7 dengan D=50



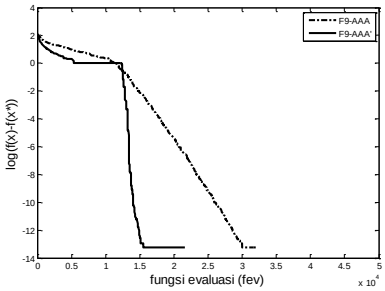
Gambar 28. Grafik laju konvergensi F8 dengan D=10



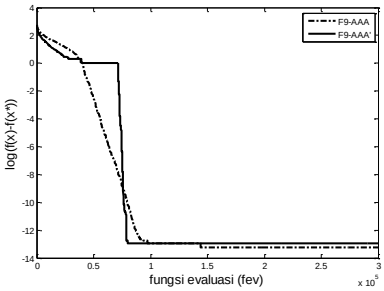
Gambar 29. Grafik laju konvergensi F8 dengan D=30



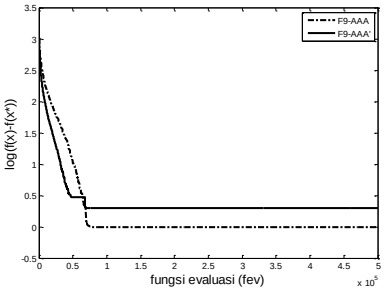
Gambar 30. Grafik laju konvergensi F8 dengan D=50



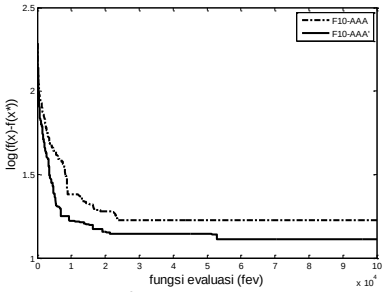
Gambar 31. Grafik laju konvergensi F9 dengan D=10



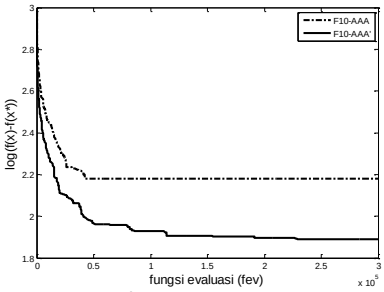
Gambar 32. Grafik laju konvergensi F9 dengan D=30



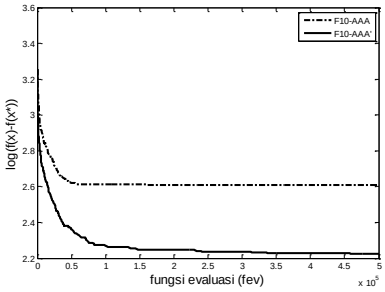
Gambar 33. Grafik laju konvergensi F9 dengan D=50



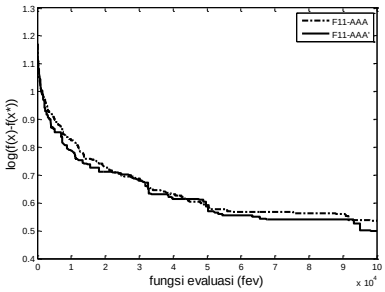
Gambar 34. Grafik laju konvergensi F10 dengan D=10



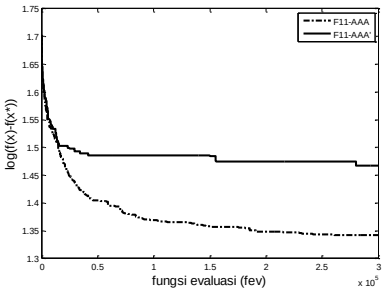
Gambar 35. Grafik laju konvergensi F10 dengan D=30



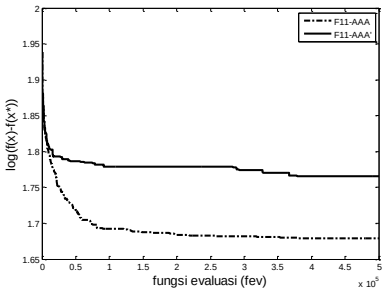
Gambar 36. Grafik laju konvergensi F10 dengan D=50



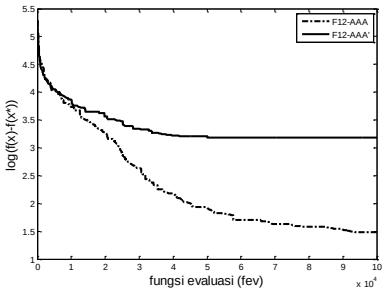
Gambar 37. Grafik laju konvergensi F11 dengan D=10



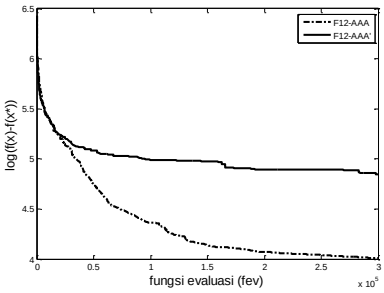
Gambar 38. Grafik laju konvergensi F11 dengan D=30



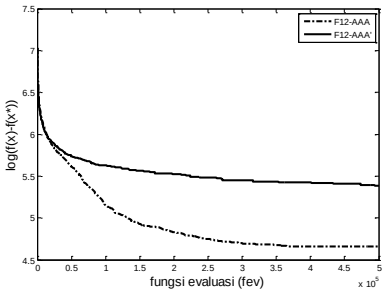
Gambar 39. Grafik laju konvergensi F11 dengan D=50



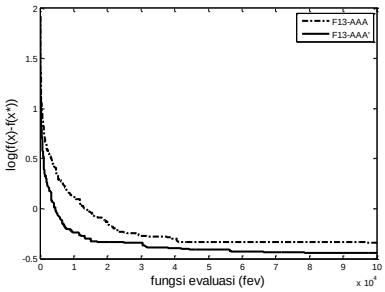
Gambar 40. Grafik laju konvergensi F12 dengan D=10



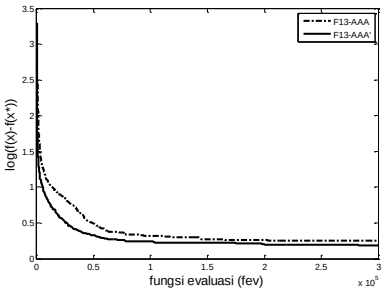
Gambar 41. Grafik laju konvergensi F12 dengan D=30



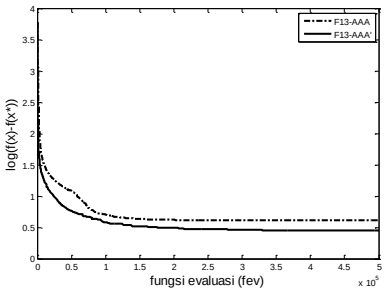
Gambar 42. Grafik laju konvergensi F12 dengan D=50



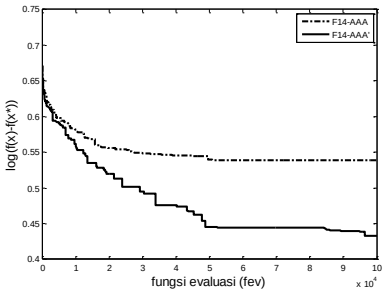
Gambar 43. Grafik laju konvergensi F13 dengan D=10



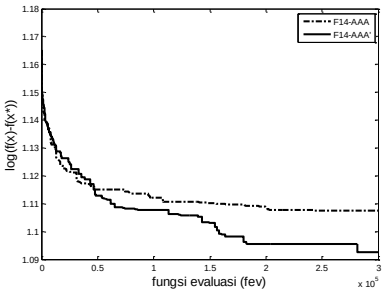
Gambar 44. Grafik laju konvergensi F13 dengan D=30



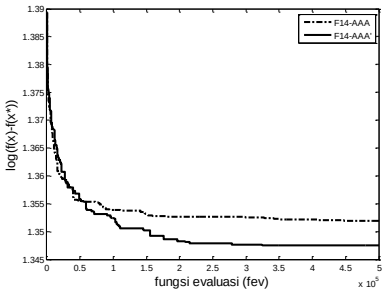
Gambar 45. Grafik laju konvergensi F13 dengan D=50



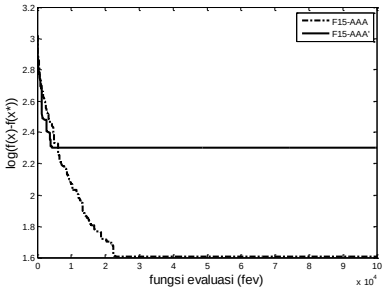
Gambar 46. Grafik laju konvergensi F14 dengan D=10



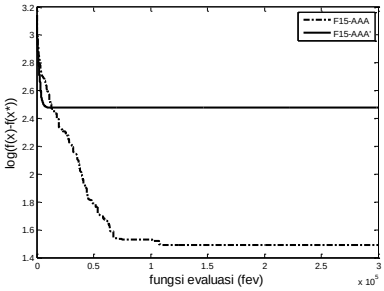
Gambar 47. Grafik laju konvergensi F14 dengan D=30



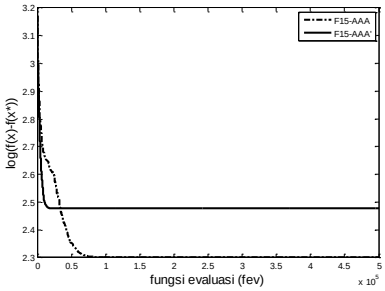
Gambar 48. Grafik laju konvergensi F14 dengan D=50



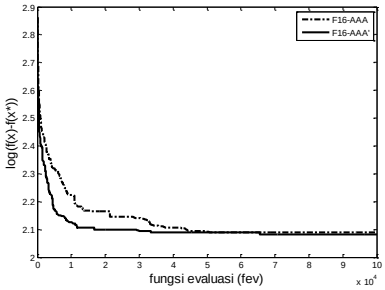
Gambar 49. Grafik laju konvergensi F15 dengan D=10



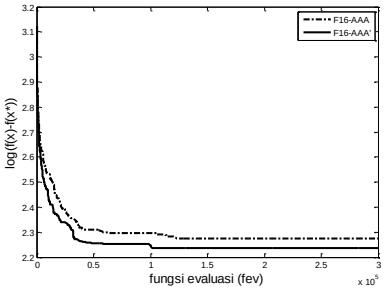
Gambar 50. Grafik laju konvergensi F15 dengan D=30



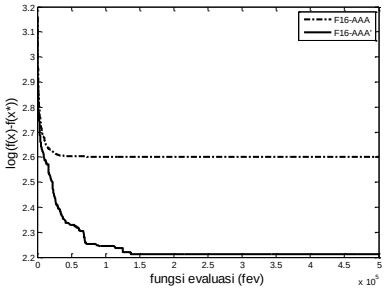
Gambar 51. Grafik laju konvergensi F15 dengan D=50



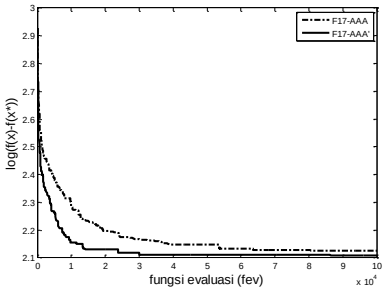
Gambar 52. Grafik laju konvergensi F16 dengan D=10



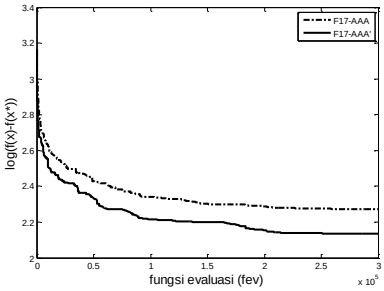
Gambar 53. Grafik laju konvergensi F16 dengan D=30



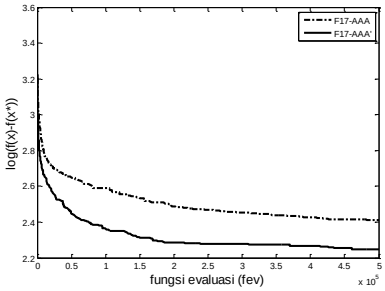
Gambar 54. Grafik laju konvergensi F16 dengan D=50



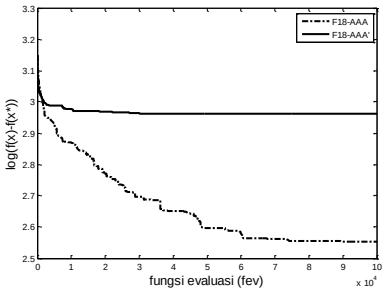
Gambar 55. Grafik laju konvergensi F17 dengan D=10



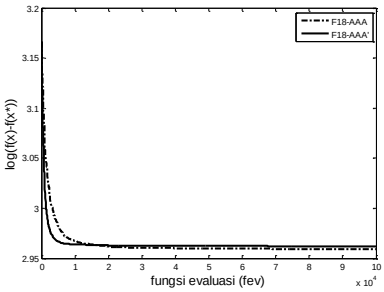
Gambar 56. Grafik laju konvergensi F17 dengan D=30



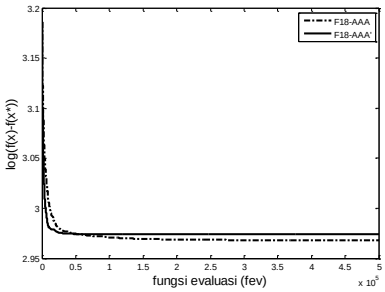
Gambar 57. Grafik laju konvergensi F17 dengan D=50



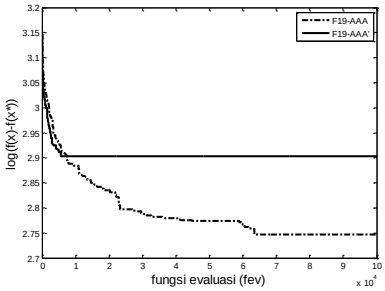
Gambar 58. Grafik laju konvergensi F18 dengan D=10



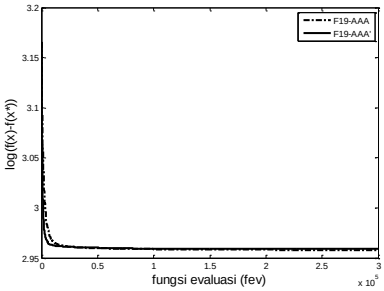
Gambar 59. Grafik laju konvergensi F18 dengan D=30



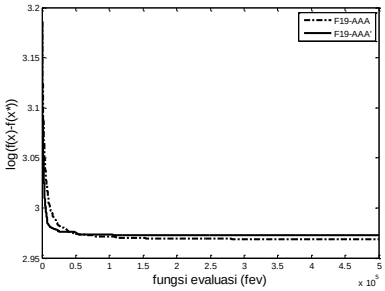
Gambar 60. Grafik laju konvergensi F18 dengan D=50



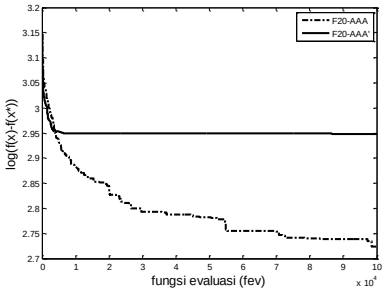
Gambar 61. Grafik laju konvergensi F19 dengan D=10



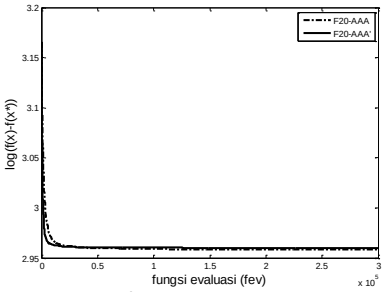
Gambar 62. Grafik laju konvergensi F19 dengan D=30



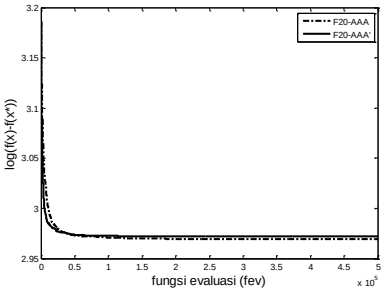
Gambar 63. Grafik laju konvergensi F19 dengan D=50



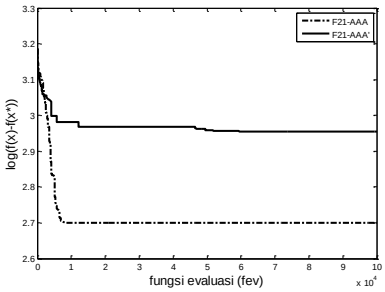
Gambar 64. Grafik laju konvergensi F20 dengan D=10



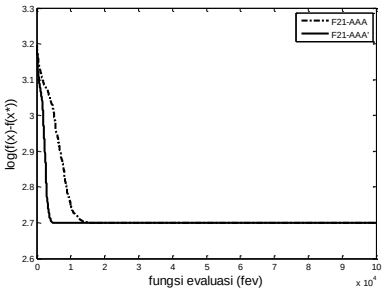
Gambar 65. Grafik laju konvergensi F20 dengan D=30



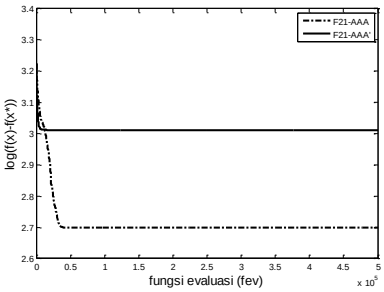
Gambar 66. Grafik laju konvergensi F20 dengan D=50



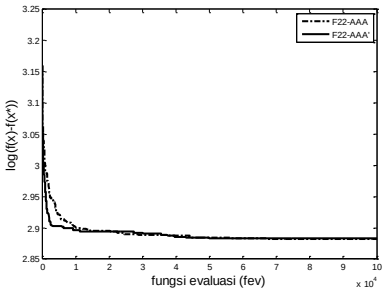
Gambar 67. Grafik laju konvergensi F21 dengan D=10



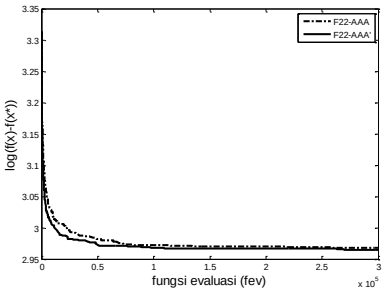
Gambar 68. Grafik laju konvergensi F21 dengan D=30



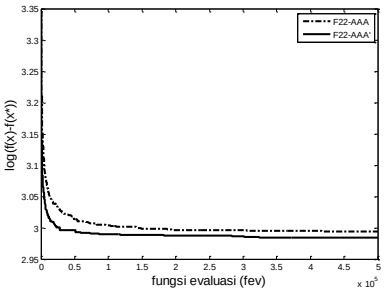
Gambar 69. Grafik laju konvergensi F21 dengan D=50



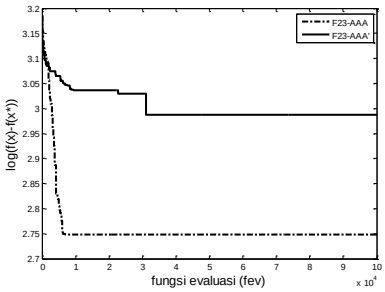
Gambar 70. Grafik laju konvergensi F22 dengan D=10



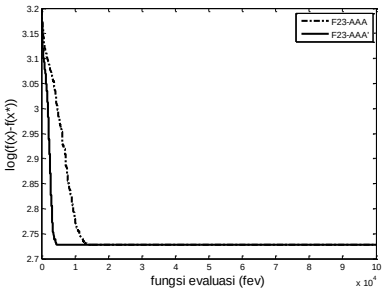
Gambar 71. Grafik laju konvergensi F22 dengan D=30



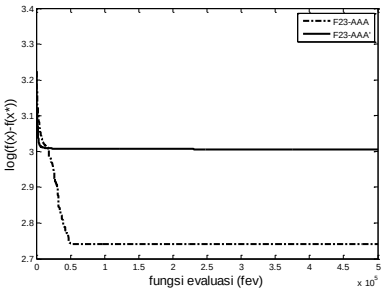
Gambar 72. Grafik laju konvergensi F22 dengan D=50



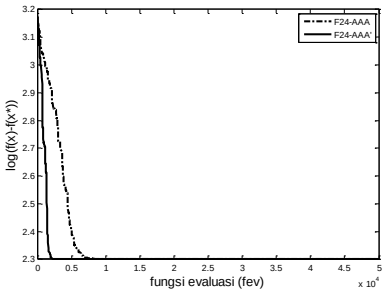
Gambar 73. Grafik laju konvergensi F23 dengan D=10



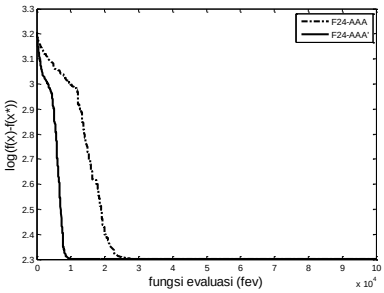
Gambar 74. Grafik laju konvergensi F23 dengan D=30



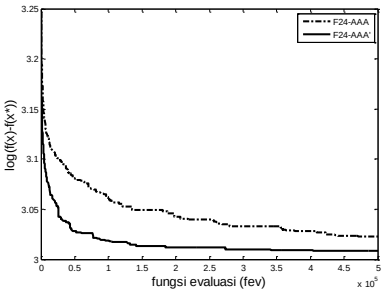
Gambar 75. Grafik laju konvergensi F23 dengan D=50



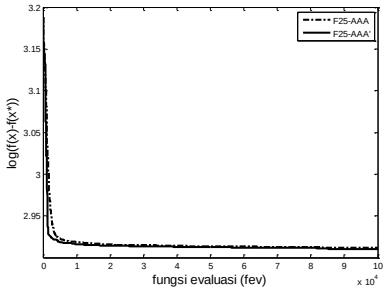
Gambar 76. Grafik laju konvergensi F24 dengan D=10



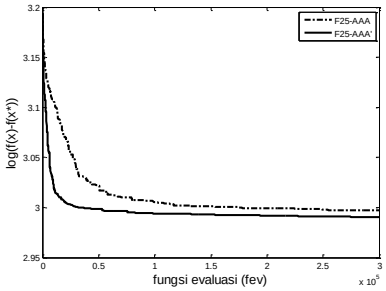
Gambar 77. Grafik laju konvergensi F24 dengan D=30



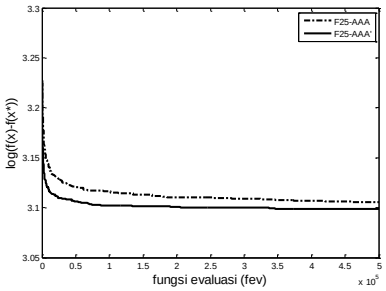
Gambar 78. Grafik laju konvergensi F24 dengan D=50



Gambar 79. Grafik laju konvergensi F25 dengan D=10



Gambar 80. Grafik laju konvergensi F25 dengan D=30



Gambar 81. Grafik laju konvergensi F25 dengan D=50

Kompleksitas algoritma menentukan seberapa rumit sebuah algoritma sehingga berpengaruh pada waktu eksekusi algoritma tersebut. Tabel 7 adalah perbandingan kompleksitas algoritma antara AAA dan AAA' pada dimensi 10, 30, dan 50.

Tabel 7. Perbandingan Waktu Eksekusi AAA dan AAA' pada Dimensi 10, 30, dan 50 Berdasarkan CEC'05

D	T0	T1	T2		(T2-T1)/T0	
			AAA	AAA'	AAA	AAA'
10	0.3663113	30.18960	43.57756	40.38455	36.54804	27.83138
30		35.22952	48.93445	46.56413	37.41334	30.94256
50		41.50146	54.51078	51.76396	35.51438	28.01579

T0 adalah waktu eksekusi program sesuai pada panduan CEC'05. Untuk meningkatkan akurasi, nilai T0 adalah rata-rata dari 500 kali eksekusi. T1 adalah waktu eksekusi fungsi F3 sebanyak 200000 kali sesuai dimensi D. Untuk meningkatkan akurasi, nilai T1 pada dimensi yang berbeda diambil rata-rata dari 100 kali eksekusi. T2 adalah waktu eksekusi algoritma AAA atau AAA' untuk menyelesaikan fungsi F3 dengan maksimum evaluasi 200000 pada setiap dimensi D. Untuk meningkatkan akurasi, nilai T2 adalah rata-rata dari 100 kali eksekusi algoritma pada setiap dimensi D.

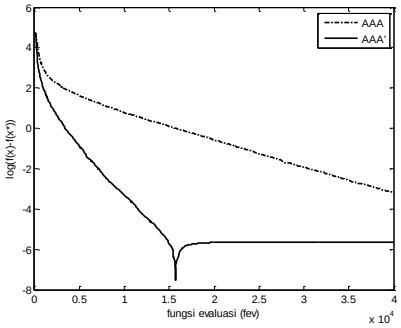
Nilai (T2-T1)/T0 menunjukkan seberapa kompleks algoritma tersebut jika ditinjau dari segi waktu. Semakin tinggi nilai (T2-T1)/T0 menandakan semakin lama waktu yang dibutuhkan untuk mengeksekusi algoritma tersebut. Berdasarkan Tabel 7, waktu eksekusi AAA' lebih kecil dari AAA yang berarti proses pada AAA lebih kompleks dari AAA' jika ditinjau dari waktu eksekusi.

Algoritma AAA memiliki tiga parameter, yaitu jumlah koloni alga (M), jumlah individu pada tiap koloni (N), dan jumla maksimum evaluasi (O). Terdapat empat fungsi pendukung dalam perhitungan AAA, yaitu:

a) *CalculateGreatness()*, fungsi ini untuk menghitung kebugaran koloni dalam populasi. Fungsi ini memiliki kompleksitas *O*(M).
b) *GreatnessOrder()*, fungsi ini untuk mengurutkan tingkat kebugaran koloni alga berdasarkan parameter tertentu. Kompleksitas untuk fungsi ini yaitu $O(M^2)$.
c) *FrictionSurface()*, fungsi ini untuk menghitung gaya gesek pada permukan cairan yang dialami oleh alga. Kompleksitas untuk fungsi ini adalah *O*(M).
d) *TournamentSelection()*, fungsi untuk memilih koloni terbaik tetangga (*local best*) dengan teknik tournament selection. Kompleksitas untuk fungsi ini yaitu *O*(M).

Berdasarkan perhitungan tersebut, AAA memiliki kompleksitas $O(M^3N^2O)$ sedangkan AAA' memiliki kompleksitas $O(M^2N^2O)$. Proses yang membuat AAA lebih kompleks dari AAA' yaitu proses *tournamen selection*. Pada AAA proses *tournament selection* dilakukan setiap koloni bergerak, sedangkan pada AAA' proses tersebut hanya dilakukan ketika solusi lebih buruk dari sebelumnya.

Pada implementasi optimasi desain pressure vessel, eksekusi sebanyak 500 kali dengan *random seed* 1-500 dan masing-masing eksekusi maksimal 40000 kali fungsi evaluasi (fev). Hasil pengujian tersebut menghasilkan laju konvergensi seperti pada Gambar 82.



Gambar 82. Laju konvergensi AAA dan AAA' pada kasus pressure vessel dengan maksimum 40000 fungsi evaluasi

Grafik laju konvergensi menyatakan perbandingan semilog selisih antara median hasil *f(x)* dan target *f(x*)* terhadap fungsi evaluasi. Grafik pada Gambar 82 menunjukkan laju konvergensi AAA' lebih baik dari AAA karena pada fev=$1.6x10^4$ sudah terjadi konvergensi. Detail hasil dan waktu eksekusi pada fungsi evaluasi 2000 dan 40000 tampak pada Tabel 8. Tabel 9 menunjukkan parameter $x_{1\text{-}4}$, konstanta $y_{1\text{-}2}$, *constrain* $g_{1\text{-}6}$, dan hasil *f(x)* terbaik, sedangkan Tabel 10 menunjukkan perbandingan waktu perolehan solusi pada kondisi terbaik AAA dan AAA'.

Berdasarkan Tabel 7 dan 10, pada jumlah fungsi evaluasi yang sama, AAA' 1.103 kali lebih cepat dari AAA. Hal ini disebabkan proses *tournamen selection* pada AAA dilakukan setiap gerakan heliks, sedangkan pada AAA' dilakukan jika solusi tidak lebih baik setelah gerakan heliks. Pada kondisi terbaik, AAA' lebih cepat 4.921 kali dalam menemukan solusi. Sedangkan pada kondisi terburuk, AAA' terjebak dalam *local optima* yaitu pada 7903.67564. Hal ini disebabkan AAA' terlalu kuat dalam mengeksploitasi kandidat solusi (*global best*) yang belum tentu sebagai solusi terbaik (*global optima*). Namun demikian, AAA' memiliki laju konvergensi yang lebih baik dari AAA pada kasus optimasi desain *pressure vessel*.

Tabel 8 Perbandingan Hasil AAA dan AAA' untuk Kasus *Pressure Vessel* pada Fungsi Evaluasi 20000 Dan 40000

	Algoritma	**20000**	**40000**
Solusi			
Terbaik	AAA	7197.73281	**7197.72893**
	AAA'	**7197.72893**	**7197.72893**
Terburuk	AAA	7289.74786	7262.24008
	AAA'	7903.67564	7903.67564
Rata-rata	AAA	7199.26473	7198.00775
	AAA'	7205.05995	7204.84851
Std	AAA	6.0051502	3.3213675
	AAA'	57.5932321	57.5197376
Waktu Eksekusi			
Rata-rata	AAA	5.0486	10.1295
	AAA'	**4.5748**	**9.1831**
Std	AAA	0.0301211	0.0900636
	AAA'	0.0416411	0.0988414

Tabel 9. Perbandingan Parameter $X_{1\text{-}4}$, Konstanta $Y_{1\text{-}2}$, Constrain $G_{1\text{-}6}$, dan Hasil F(X) Antara AAA Dan AAA' Pada Kasus Desain *Pressure Vessel* dengan Fungsi Evaluasi 20000 Dan 40000

Parameter & Target	Algoritma	20000	40000
x_1 (Ts) = 1.125	AAA	1.12925569	1.125
	AAA'	1.12609603	1.12501047
x_2 (Th) = 0.625	AAA	0.64781226	0.63721248
	AAA'	0.62547076	0.625
x_3 (R) = 58.2901554	AAA	58.2901166	58.2901554
	AAA'	58.2901554	58.2901554
x_4 (L) = 43.6926562	AAA	43.6929034	43.6926565
	AAA'	43.6926562	43.6926562
$y_1 = 1.25$	AAA	1.125	1.125
	AAA'	1.125	1.125
$y_2 = 0.625$	AAA	0.625	0.625
	AAA'	0.625	0.625
$g_1 \leq 0$	AAA	-0.004256	-0.000000
	AAA'	-0.001096	-0.000010
$g_2 \leq 0$	AAA	-0.094056	-0.083456
	AAA'	-0.071714	-0.071244
$g_3 \leq 0$	AAA	-0.357398	-0.000764
	AAA'	-0.000000	0.000000
$g_4 \leq 0$	AAA	-196.307097	-196.307343
	AAA'	-196.307344	-196.307344
$g_5 \leq 0$	AAA	-0.029256	-0.025000
	AAA'	-0.026096	-0.025010
$g_6 \leq 0$	AAA	-0.047812	-0.037212
	AAA'	-0.025471	-0.025000
$f(x) = 7197.72893$	AAA	7197.73281	**7197.72893**
	AAA'	**7197.72893**	**7197.72893**

Tabel 10. Perbandingan Waktu Perolehan Solusi Terbaik Berrdasarkan *Ramdom Seed* dan Fungsi Evalusasi AAA dan AAA'

Algoritma	*Random Seed*	Fungsi Evaluasi	Waktu (s)
AAA	185	38200	9.626130
AAA'	247	8240	1.956132

4. KESIMPULAN

Artificial Algae Algorithm (AAA) merupakan algoritma optimasi dengan model *swarm* yaitu gerakan heliks dan model evolusi yaitu dengan reproduksi dan adaptasi. Pada penelitian ini dilakukan optimasi terhadap gerakan heliks untuk meningkatkan laju konvergensi, khususnya pada kasus optimasi desain *pressure vessel*. Optimasi yang dilakukan yaitu menggerakkan alga menuju koloni terbaik dalam populasi (*global best*).

Berdasarkan pengujian CEC'05, AAA' mengalami peningkatan laju konvergensi namun terjadi penurunan kestabilan algoritma. Hal tersebut dapat dianalisis berdasarkan beberapa indikator berikut:

a) Dalam kondisi terbaik, AAA' lebih cepat menemukan solusi. Namun dalam kondisi terburuk, AAA' terjebak dalam *local optima*.
b) Seiring peningkatan dimensi, kualitas hasil cenderung fluktuatif. Pada dimensi 10, AAA' mampu meyelesaikan 11 fungsi lebih baik dari AAA. Pada dimensi 30, kemampuan AAA' menurun sehingga

mampu menyelesaiakn 8 fungsi lebih baik dari AAA. Sedangkan pada dimensi 50, AAA' kembali menyelesaikan 11 fungsi lebih baik dari AAA.

c) Kompleksitas waktu (*running time*) pada AAA' turun menjadi 76.15% dari AAA, pada dimensi 10 turun menjadi 82.70% dari AAA, dan pada dimensi 50 turun menjadi 78.89% dari AAA.
d) Kompleksitas algoritma AAA adalah $O(M^3N^2O)$ dan AAA' adalah $O(M^2N^2O)$.

Pada kasus *pressure vessel* dengan jumlah fungsi evaluasi 20000 dan 40000, AAA' 1.103 kali lebih cepat dari AAA. Sedangkan pada kasus terbaik, untuk menemukan solusi AAA' 4.921 kali lebih cepat dari AAA. Pada kasus terburuk, AAA' terjebak dalam *local optima*. Hal ini disebabkan gerakan heliks terlalu fokus pada *global best* yang belum tentu adalah *global optima*. Dengan demikian, penelitian ini dapat dilanjutkan dengan meningkatkan eksplorasi terhadap kandidat solusi selain *global best*, sehingga AAA' memiliki laju konvergensi yang baik tanpa terjebak pada *local optima*.

5. REFERENSI

[1] Taqiyuddin dan Sasongko P.H. 2013. *Studi Optimal Power Flow pada Sistem Kelistrikan 500 kV Jawa-Bali dengan Menggunakan Particle Swarm Optimization (PSO)*. Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI) Vol. 2 No. 3.

[2] A.S. Sinaga. 2014. *Pembebanan Ekonomis dengan Pengendalian Emisi pada Pembangkit Termis Menggunakan Algoritma Evolusi Diferensial*. Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI) Vol.3 No.2.

[3] A.A. Aburomman & Mamun Bin Ibne R. 2016. *A Novel SVM-kNN-PSO Ensemble Method For Intrusion Detection System*. Applied Soft Computing Vol.38, hlm. 360-372.

[4] M. Ranjani & P. Murugesan. 2015. *Optimal Fuzzy Controller Parameters Using PSO For Speed Control of Quasi-Z Source DC/DC Converter Fed Drive*. Applied Soft Computing Vol.27, hlm. 332-356.

[5] Ruhul A. Sarker & Charles S. Newton. 2008. *Optimization Modelling : A Practival Approach*. Boca Raton, CRC Press.

[6] Binitha S. & S.S. Sathya. 2012. *A Survey of Bio inspired Optimization Algorithms*. International Journal of Soft Computing and Engineering (IJSCE), Vol,2, Issue-2.

[7] K.T. Meetei. 2014. *A Survey: Swarm Intelligence vs. Genetic Algorithm*. International Journal of Science and Research (IJSR), hlm. 231-235.

[8] Rashmi A. Mahale & S.D.Chavan. 2012. *A Survey: Evolutionary and Swarm Based Bio-Inspired Optimization Algorithms*. International Journal of Scientific and Research Publications Vol.2, Issue 12.

[9] Xin-She Yang, 2014. *Swarm Intelligence Based Algorithms: A Critical Analysis*. Evolutionary Intelligence Vol.7, hlm. 17-28.

[10] Millie Pant & Radha Thangaraj. 2007. *A New Particle Swarm Optimization with Quadratic Crossover*. International Conference of Advanced Computing and Communications (ADCOM), hlm. 81–86.

[11] Sabine Helwig, Frank Neumann, dan Rolf Wanka. 2011. *Particle Swarm Optimization with Velocity Adaptation*. Handbook of Swarm Intelligence Vol.8 of the series Adaptation, Learning, and Optimization, hal. 155-173.

[12] Swagatam Das & Ajith Abraham. 2006. *Synergy of Particle Swarm Optimization with Evolutionary Algorithms for Intelligent Search and Optimization*. Proceedings of IEEE International Congress on Evolutionary Computation Vol.1, hlm. 84-88.

[13] Kedar N.D. & Raghav P.P. 2014. *Synergy of Differential Evolution and Particle Swarm Optimization*. Proceedings of the Third International Conference on Soft Computing for Problem Solving Vol.258 of the series Advances in Intelligent Systems and Computing, hlm. 143-160.

[14] Sait Ali Uymaz, Gulay Tezel, & Esra Yel. 2015. *Artificial Algae Algorithm (AAA) for nonlinear global optimization*, Applied Soft Computing Vol.31, hlm. 153-171.

[15] G.C. Onwubolu & B.V. Babu. 2004. *New Optimization Techniques in Engineering*. Springer, Berlin. Germany.